

A decorative graphic on the left side of the slide consisting of overlapping geometric shapes. It includes a blue parallelogram and a yellow parallelogram, both tilted at an angle, set against a dark blue background with subtle diagonal lines.

saschavanessen@gmail.com

Python Course

Week 1:
Writing your first program

(voor vrouwen)





About Me

- Sascha van Essen
- Bachelor Media and Culture
- Finishing Master Information Studies
- Teaching programming (to students)
- I like...
 - going to stand-up comedy
 - baking
 - reading
 - learning new things by creating applications





Workshop Overview

1. Writing your first program
2. Making choices and reuse code
3. Loops and strings
4. Files and lists
5. Dictionaries and tuples



Acknowledgements

- Charles Severance aka Dr. Chuck
 - Course “Programming for Everybody”
 - Public material available
 - Structure of this course
 - Examples of programs



Why SHOULDN'T you learn?

- Takes years (to program really well)
- “Coding is not a goal, it’s a tool for solving problems”
- Programming doesn’t make you rich
- There are always people better than you

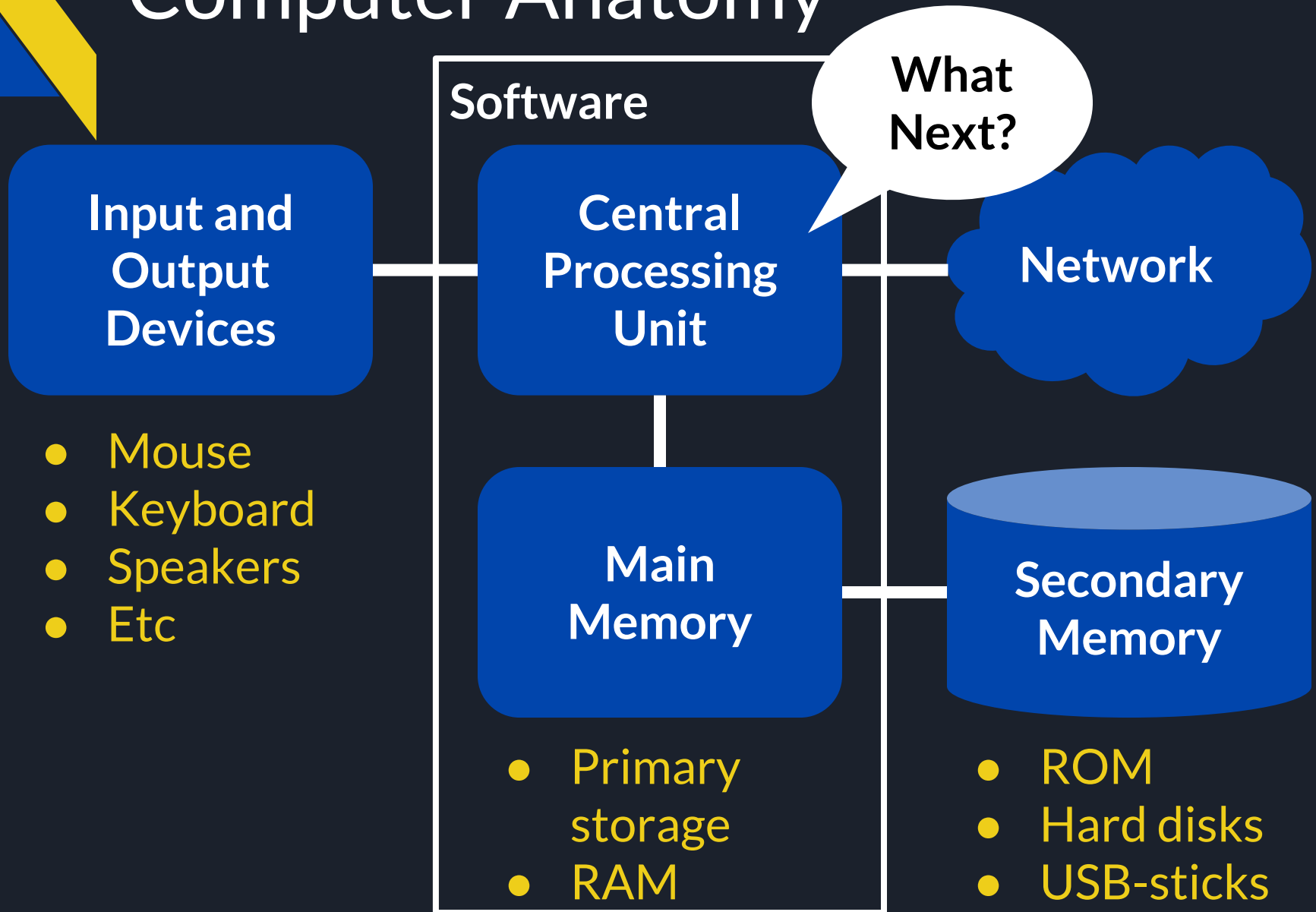
<https://www.fastcompany.com/3020126/no-you-dont-need-to-learn-to-code>



Why SHOULD you learn?

- It's fun and creative
- Because you can
- It's a useful tool
 - Science, apps, games, Internet of Things, etc
- A computer is fast and good in repetitive tasks
- There is no existing program for your needs

Computer Anatomy

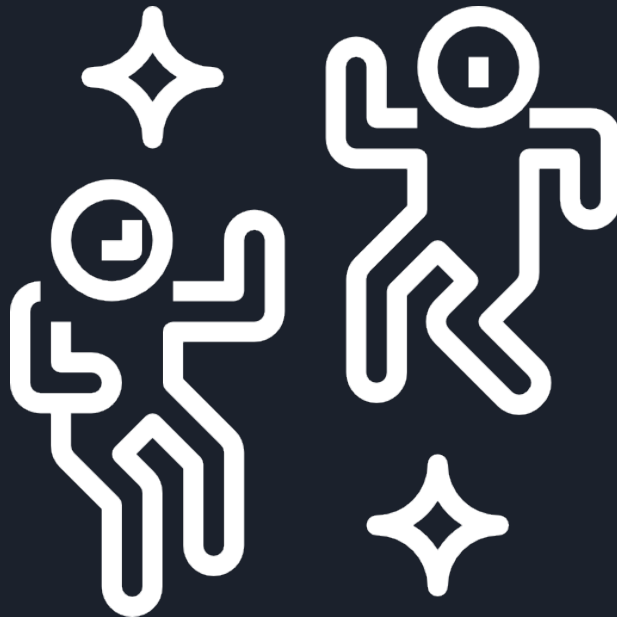




Programming Language

- Give instructions to the CPU
- Like a recipe or a dance choreography...

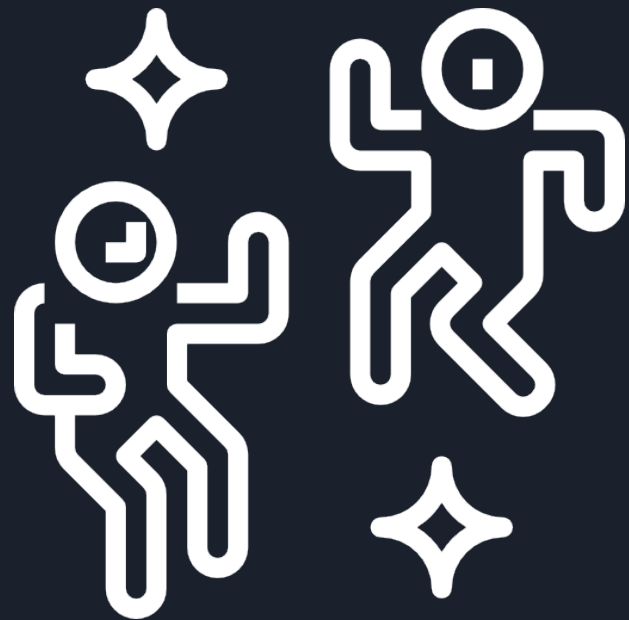
https://youtu.be/nKO_32134dU?t=9s



Programming Language

While music is playing:

- Left hand out and up
- Right hand out and up
- Flip left hand
- Flip right hand
- Left hand to right shoulder
- Right hand to right shoulder
- Left hand to back of head
- Right **ham** to back of head
- Left hand to right **hik**
- Right hand to left **hik**
- Left hand on left bottom
- Right hand on right bottom
- Wiggle
- Wiggle
- Jump





Computer Language

0010111010011001
0101100000011110
1010111010100100
1010000111010111
1111010101010011
1010101001010101
0010100010001011
0011010010011101
0001101100101110

```
file = open(filename, 'r')  
song = file.read()  
words = song.split()  
counts = dict()
```

```
for word in words:  
    counts[word] = counts.get(word, 0) + 1
```



Why Python

- Easy to learn
- Very readable
 - Easy constructs
 - No special cryptic characters
 - Forced indentation
- Used a lot -> much code available
- Cross-platform

<http://helloworldcollection.de/>



How smart is a computer?

- A computer is not smart
 - Limited vocabulary
 - It needs literal instructions
- It will stop and “complain” when instructions are not clear
 - Usually a hint to guide you to the problem
- For example:
<https://youtu.be/KUB-aJXquUA?t=19m10s>

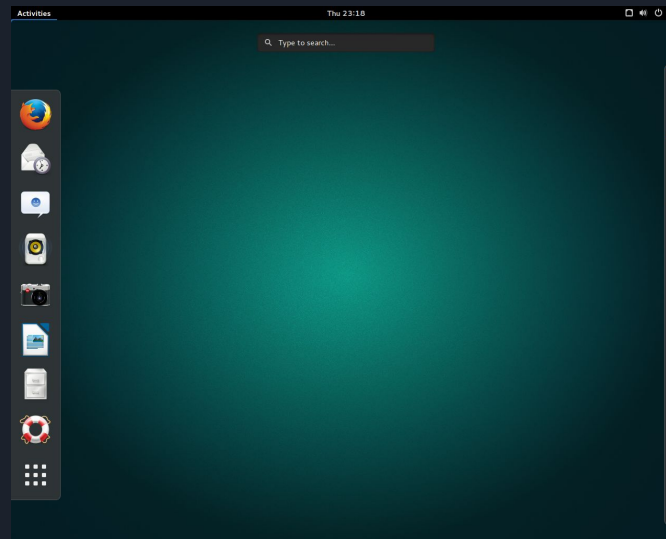
Terminal, scripts, Python?



Windows



Mac



Linux

Behind all that...

```
Command Prompt
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Administrator>chkdsk
The type of the file system is NTFS.

WARNING! F parameter not specified.
Running CHKDSK in read-only mode.

CHKDSK is verifying files (stage 1 of 3)...
File verification completed.
CHKDSK is verifying indexes (stage 2 of 3)...
Index verification completed.
CHKDSK is verifying security descriptors (stage 3 of 3)...
Security descriptor verification completed.
CHKDSK is verifying Usn Journal...
Usn Journal verification completed.

16609288 KB total disk space.
15005300 KB in 85352 files.
57336 KB in 10426 indexes.
0 KB in bad sectors.
209688 KB in use by the system.
65536 KB occupied by the log file.
1336964 KB available on disk.

4096 bytes in each allocation unit.
4152322 total allocation units on disk.
334241 allocation units available on disk.

C:\Documents and Settings\Administrator>
```

Windows

This is called a:

- Terminal
- Command-line
- Shell

```
Terminal
-rwxr-xr-x 1 sys 52850 Jun 8 1979 hptmunix
drwxrwxr-x 2 bin 320 Sep 22 05:33 lib
drwxrwxr-x 2 root 96 Sep 22 05:46 mdec
-rwxr-xr-x 1 root 50990 Jun 8 1979 rkunix
-rwxr-xr-x 1 root 51982 Jun 8 1979 r12unix
-rwxr-xr-x 1 sys 51790 Jun 8 1979 rphtunix
-rwxr-xr-x 1 sys 51274 Jun 8 1979 rptmunix
drwxrwxrwx 2 root 48 Sep 22 05:50 tmp
drwxrwxr-x12 root 192 Sep 22 05:48 usr
# ls -l /usr
total 11
drwxrwxr-x 3 bin 128 Sep 22 05:45 dict
drwxrwxrwx 2 dmr 32 Sep 22 05:48 dmr
drwxrwxr-x 5 bin 416 Sep 22 05:46 games
drwxrwxr-x 3 sys 496 Sep 22 05:42 include
drwxrwxr-x10 bin 528 Sep 22 05:43 lib
drwxrwxr-x11 bin 176 Sep 22 05:45 man
drwxrwxr-x 3 bin 208 Sep 22 05:46 mdec
drwxrwxr-x 2 bin 80 Sep 22 05:46 pub
drwxrwxr-x 6 root 96 Sep 22 05:45 spool
drwxrwxr-x13 root 208 Sep 22 05:42 src
# ls -l /usr/dmr
total 0
#
```

Linux

```
saschaessen ~ -bash - 80x24
Mach Virtual Memory Statistics: (page size of 4096 bytes)
Pages free: 14663.
Pages active: 475807.
Pages inactive: 455746.
Pages speculative: 1195.
Pages throttled: 0.
Pages wired down: 535015.
Pages purgeable: 26296.
"Translation faults": 580888111.
Pages copy-on-write: 12237853.
Pages zero filled: 312498781.
Pages reactivated: 67977823.
Pages purged: 21020288.
File-backed pages: 194142.
Anonymous pages: 738606.
Pages stored in compressor: 2650385.
Pages occupied by compressor: 614492.
Decompressions: 85707470.
Compressions: 119039843.
Pageins: 20820796.
Pageouts: 308549.
Swapins: 16433866.
Swapouts: 18178224.

macbook-pro:~ saschaessen$
```

Mac

- Interface
 - No fancy windows
 - No icons
 - No clicking
- You need to type in commands to the computer

Directories and Files

The image shows two overlapping windows. The left window is a terminal titled 'Pythoncursus — -bash — 55x20'. It displays the following text:

```
Last login: Mon May 28 17:23:39 on ttys003
MacBook-Pro:~ saschaessen$ cd documents/pythoncursus
MacBook-Pro:pythoncursus saschaessen$ ls
helloworld.py  myscript.py  pythonlearn.pdf
MacBook-Pro:pythoncursus saschaessen$
```

The right window is a macOS Finder titled 'pythoncursus'. It shows a sidebar with 'Favorieten' (Favorites) and 'Apparaten' (Devices). The main pane shows a table of files in the 'pythoncursus' directory:

Naam	Bewerkingsdatum	Grootte
helloworld.py	Eergisteren 15:50	55 bytes
myscript.py	Vandaag 17:23	33 bytes
pythonlearn.pdf	Vandaag 17:21	2,4 MB

- `cd [directoryname]` -> go to the specified directory (Change Directory)
- `dir` -> show content of directory, Windows
- `ls` -> show content of directory, Linux/Mac



Starting Terminal/Cmd/Shell

Een programma op je computer

- Windows: search for *cmd*
- Mac: search for *terminal*
- Linux: *Ctrl-Alt-T*



Starting Python

Start a terminal window

```
$ python
Python 2.7.10 (default, Oct  6 2017, 22:29:07)
[GCC 4.2.1 Compatible Apple LLVM 9.0.0 (clang-900.0.31)]
on darwin
Type "help", "copyright", "credits" or "license" for
more information.
>>>
```



\$ to signify that we are doing something in the terminal

White text you should type yourself

Yellow text you should see in the terminal



Python: First Attempt

```
>>> Hi there
```

```
SyntaxError: invalid syntax
```

```
>>> answer me!
```

```
SyntaxError: invalid syntax
```

```
>>> help
```

```
Type help() for interactive help, or help(object) for  
help about object.
```

```
>>> help()
```

```
Welcome to Python 2.7! This is the online help utility.
```

```
...
```

```
help>
```



Python: help()

```
help> keywords
```

```
Here is a list of the Python keywords.  Enter any  
keyword to get more help.
```

```
...
```

```
help> if
```

```
The "if" statement is used for conditional execution:
```

```
...
```

```
(END) q
```

```
help> quit
```

```
You are now leaving help and returning to the Python  
interpreter.
```

```
...
```



Hello world

```
>>> print("Hello world")
```

```
...
```

```
>>> 27 + 15
```

```
...
```

```
>>> x = 7
```

```
>>> print(x)
```

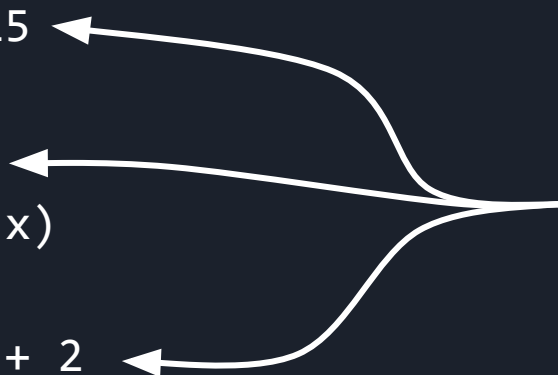
```
...
```

```
>>> x = x + 2
```

```
>>> print(x)
```

```
...
```

```
>>>
```



Spaces are not necessary,
but they make your script clearer



Booleans

```
>>> 1 + 2 = 3
```

```
SyntaxError: can't assign to operator
```

```
>>> 1 + 2 == 3
```

```
True
```

```
>>>
```

`x = 3` `x` becomes 3

`x == 3` is `x` 3?



Text editor



- To make (Python) scripts
- Open Sublime, Notepad++, or other editor
- Open new file (*Ctrl-N / Cmd-N*)
- Save file in *Documents*, in a new folder you create named *pythoncursus*
- Name the file *myscript.py*
- Do not use spaces in folder or file names!



Scripts

- A set of instructions, like a recipe
- Type the following in *myscript.py*
- Save the script (Ctrl-S / Cmd-S)

```
x = 7  
print(x)  
x = x + 2  
print(x)
```




Execute scripts

Go to the terminal

```
>>> quit()
```

or Ctrl-D

```
$ cd documents/pythoncursus
```

```
$ dir (windows)           $ ls (mac/linux)
```

```
myscript.py
```

```
$ python myscript.py
```

```
...
```



Hello <your-name-here>

- Make a new script called *helloworld.py*

```
print("Enter your name:")  
name = input()  
print("Hello", name)
```

- Start script from terminal

```
$ python helloworld.py  
Enter your name:  
...  
Hello ...
```





Data Types

- Start an interactive Python session
- The basic data types:

```
>>> type(4)
```

```
>>> type(1.5)
```

```
>>> type("Yo")
```

```
>>> type('Yo')
```

```
>>> type("13")
```



Variables

- “Save” a value for later
- Like storing something in a box
- You assign a value to a variable with =

```
>>> message = "hello"
```

```
>>> n = 2
```

```
>>> pi = 3.14
```

```
>>> print(message)
```

```
>>> type(message)
```

```
>>> print(n)
```

```
>>> type(n)
```

```
>>> type(pi)
```





Variable names

- A combination of letters, underscores and numbers (not in the beginning)
- No reserved words!
- No special characters!
- Case sensitive
- Use clear and meaningful names!



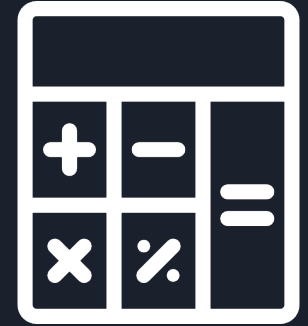


Variables names

- No:
 - 9to5 = “working hours”
 - mail@ = “saschavanessen@gmail.com”
- Yes
 - city = “Amsterdam”
 - student_nr = 1234
 - nr1 = “me”



Calculator



- Operators: + - * / ** %

```
>>> 20 + 32
```

```
>>> minutes = 60
```

```
>>> 60 * minute
```

```
>>> 10 / 6
```

! Python 3

```
>>> 10 // 6
```

! Python 3

```
>>> 5 ** 2
```

```
>>> (5 + 9) * (15 - 7)
```



Order of Operations

1. Parenthesis: $(1 + 2) * (3 + 1) = 3 * 4 = 12$
2. Exponentiation: $2 ** 1 + 1 = 2 + 1 = 3$
3. Multiplication & division: $2 * 3 - 1 = 6 - 1 = 5$
4. Addition & subtraction: $5 - 3 - 1 = 2 - 1 = 1$



String Operations

- Concatenation

```
>>> str1 = "this"  
>>> str2 = " and this"  
>>> print(str1 + str2)
```

```
>>> num1 = '10'  
>>> num2 = '15'  
>>> print(num1 + num2)
```

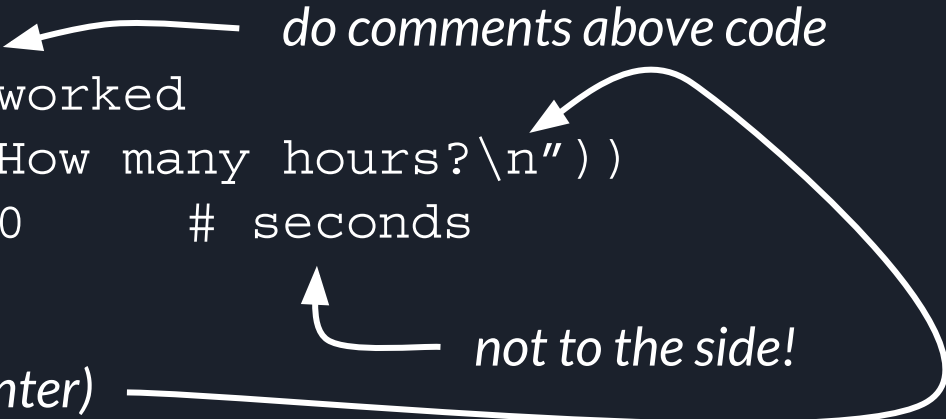
- Multiplication

```
>>> word = "bla"  
>>> n = 7  
>>> print(n * word)
```

Comments

- Scripts can get long and unreadable
- Solutions:
 - Give clear and meaningful variable names
 - Insert comments
 - Make functions

```
# Ask user for hours worked
hours = float(input("How many hours?\n"))
seconds = hours / 3600      # seconds
```



`\n` means "new line" (enter)



To be continued!

- More practice:
 - Exercises in chapter 1 and 2 of the book
- Next week: Make choices and reuse code
- See you next week!



Icons Acknowledgements

- <https://www.flaticon.com/authors/skyclick>
- <https://www.flaticon.com/authors/smashicons>
- <https://www.flaticon.com/authors/gregor-cresnar>
- <http://www.freepik.com>